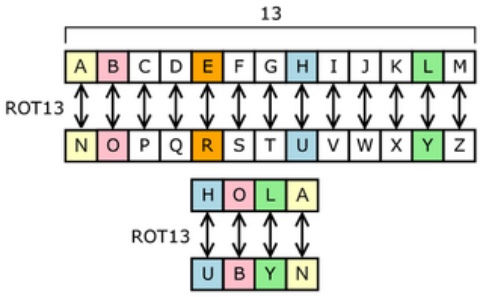
LABORATORIO 5

Programas Ada con Vectores

1. Implementa un procedimiento llamado **ultimo_par** que, dado un vector de 10 enteros, devuelva el último par de esa secuencia y la posición en la que se encuentra. Si no hubiera enteros pares la posición será 0 (el número nos da igual). Así, para (8,4,6,7,5,3,2,1,1,9) y 1 debe devolver 2 posición 7; y para (1,3,5,7,9,11,13,15,17,19) debe devolver posición 0 (para el número sería válido cualquier valor).
2. Implementa una función llamada **múltiplos_del_último** que, dado un vector de 10 enteros, devuelva cuántos de esos valores son múltiplos del último. Así, para (8,4,6,7,5,3,2,1,1,2) debe devolver 5; y para (1,3,5,7,9,11,13,15,17,19) debe devolver 1 (un número siempre es múltiplo de sí mismo).
3. Implementa la función **está_ordenado** que, dado un vector de 10 enteros, devuelva true si está ordenado creciente o decrecientemente. Así, para (1,2,3,4,5,6,7,8,9,10) debe devolver *true*; para (9,8,8,8,7,6,5,4,2,1) debe devolver también *true*; y para (1,2,3,4,5,4,3,2,1,9) debe devolver *false*.
4. Implementa el procedimiento **separar_dígitos** que, dado un número positivo y devuelva un vector con sus dígitos (si tuviera menos de 10 cifras se rellenaría con ceros a la izquierda) y devuelva también cuántos de esos dígitos son impares. Así, para 123456 debe devolver (0,0,0,0, 1,2,3,4,5,6) y 3 (dígitos impares); para 2468024680 debe devolver (2,4,6,8,0,2,4,6,8,0) y 0.
5. Implementa la función **binario_a_decimal** que dado un vector de 10 unos y ceros, devuelva el valor decimal que le corresponde a ese número. Así, para (0,0,0,0,0,1,0,0,1,0) devuelve 18 = $1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 0*2^0$.
6. Implementa el procedimiento **rot13** que, dado un string de 10 caracteres, le aplica el cifrado *rot13. **ROT13** («**rotar 13 posiciones**») es un sencillo cifrado César utilizado para ocultar un texto sustituyendo cada letra por la letra que está trece posiciones por delante en el alfabeto. La A se convierte en N, la B se convierte en O y así hasta la M, que se convierte en Z. Luego la secuencia se invierte: la N se convierte en A, la O se convierte en B y así hasta la Z, que se convierte en M. Lo mismo ocurre con las letras minúsculas, mientras que cualquier otro carácter que haya, se deja tal cual. Este algoritmo se utiliza en foros de Internet como medio para ocultar de miradas casuales el final de un chiste, la solución a un acertijo, el destripe de una película o una historia, o algún texto ofensivo. (descripción e imagen tomadas de Wikipedia).*



7. Implementa una función **digito_control** que, dado un número de cuenta corriente de 10 dígitos representado en un vector de 10 dígitos, devuelva el dígito de control que le corresponde. El dígito de control de los números de cuenta se utiliza para evitar errores al transcribir números de cuenta en los bancos ya que algunos de sus dígitos son el resultado de un cálculo sobre el resto de los dígitos. Sobre un número de 10 dígitos, el dígito de control es el resultado de la siguiente formula (d_i representa el dígito *i* empezando a contar por la izquierda). Si el resultado fuera un número de dos cifras, se sustituye el diez por un 1 y el once por un 0. Por ejemplo: el dígito de control de 1111111111 es 0, y el de 1234567898 es 4.

$$d_{control} = 11 - \left[\sum_{i=1}^{10}\left(d_i \cdot \{2^i rem\ 11\}\right)\right] rem\ 11$$

$$d_{1111111111} = 11 - [(1 \cdot \{2\ rem\ 11\}) + (1 \cdot \{4\ rem\ 11\}) + (1 \cdot \{8\ rem\ 11\}) + (1 \cdot \{16\ rem\ 11\}) + (1 \cdot \{32\ rem\ 11\}) + (1 \cdot \{64\ rem\ 11\}) + (1 \cdot \{128\ rem\ 11\}) + (1 \cdot \{256\ rem\ 11\}) + (1 \cdot \{512\ rem\ 11\}) + (1 \cdot \{1024\ rem\ 11\})]rem\ 11 = 11 - [(1 \cdot 2) + (1 \cdot 4) + (1 \cdot 8) + (1 \cdot 5) + (1 \cdot 10) + (1 \cdot 9) + (1 \cdot 7) + (1 \cdot 3) + (1 \cdot 6) + (1 \cdot 1)]rem\ 11 = 11 - [2 + 4 + 8 + 5 + 10 + 9 + 7 + 3 + 6 + 1]rem\ 11 = 11 - 55\ rem\ 11 = 11 - 0 = 11 \Rightarrow 0\ (por\ ser\ de\ dos\ cifras)$$

$$d_{1234567898} = 11 - [(1 \cdot \{2\ rem\ 11\}) + (2 \cdot \{4\ rem\ 11\}) + (3 \cdot \{8\ rem\ 11\}) + (4 \cdot \{16\ rem\ 11\}) + (5 \cdot \{32\ rem\ 11\}) + (6 \cdot \{64\ rem\ 11\}) + (7 \cdot \{128\ rem\ 11\}) + (8 \cdot \{256\ rem\ 11\}) + (9 \cdot \{512\ rem\ 11\}) + (8 \cdot \{1024\ rem\ 11\})]rem\ 11 = 11 - [(1 \cdot 2) + (2 \cdot 4) + (3 \cdot 8) + (4 \cdot 5) + (5 \cdot 10) + (6 \cdot 9) + (7 \cdot 7) + (8 \cdot 3) + (9 \cdot 6) + (8 \cdot 1)]rem\ 11 = 11 - [2 + 8 + 24 + 20 + 50 + 54 + 49 + 24 + 54 + 8]rem\ 11 = 11 - 293\ rem\ 11 = 11 - 7 = 4 \Rightarrow 4$$

PARTE OPCIONAL

8. N es un número misterioso de 10 dígitos, *abcdefghij.* Cada uno de los dígitos es diferente y los siguientes números cumplen las siguientes propiedades:

- *a* es divisible entre 1
- *ab* es divisible entre 2
- *abc* es divisible entre 3
- *abcd* es divisible entre 4
- *abcde* es divisible entre 5
- *abcdef* es divisible entre 6
- *abcdefg* es divisible entre 7
- *abcdefgh* es divisible entre 8
- *abcdefghi* es divisible entre 9
- *abcdefghij* es divisible entre 10

Construye un programa que localice el número N. El programa debe comprobar todas las restricciones. Se ofrece una bonificación para los más eficientes (en tiempo de ejecución, en espacio de memoria y/o fecha de entrega).

NOTA: Hay un pequeño inconveniente que hace falta esquivar para resolver el ejercicio: el tipo Integer no soporta enteros de 10 cifras por lo que, a no ser que *a* (el dígito de más a la izquierda) sea 0, no se puede encontrar la solución con variables de tipo integer.

9. Implementa el procedimiento **incluir_digito_control** que, dado un número de una tarjeta de crédito de 15 dígitos representado en las 15 primeras posiciones de un vector de 16 dígitos, devuelva el dígito de control que le corresponde en la posición 16. El dígito de control de las tarjetas de crédito se calcula usando el algoritmo de *Luhn*, sobre el que puedes encontrar información en Internet y se resume como el resultado de la siguiente formula (d_i representa el dígito *i* empezando a contar por la izquierda).

$$d_{16} = 10 - \left[\sum_{i=1}^{15} f(d_i, i)\right] rem\ 10 \qquad f(x, pos) = \begin{cases} x & si\ pos\ es\ par \\ 2x & si\ pos\ es\ impar\ y\ x < 5 \\ 2x - 9 & si\ pos\ es\ impar\ y\ x \geq 5 \end{cases}$$

10. Implementa la función **es_correcta**, que dado un número de cuenta representado en un vector de 16 dígitos devuelve *true* o *false* si los dígitos del vector en ese orden son correctos.

11. Implementa el procedimiento **incluir_dígitos_control** que, dado un número de una cuenta corriente (10 dígitos), el código del país (un string de 2 caracteres mayúsculas como “ES”, “FR”, “IT”…), el número de la entidad bancaria (4 dígitos) y de sucursal (4 dígitos) añada los dígitos de control de la cuenta corriente y le anteponga el código de país y los dígitos de control de su IBAN. El resultado tiene que ser un string de la forma “PPKK BBBB SSSS CC XXXXXXXXXX” (separado con blancos para más claridad. PP hace referencia a los dos caracteres a los que se refiere el país. BBBB se refiere al código del banco. SSSS se refiere al número de sucursal dentro del banco. XXXXXXXXXX es el número de cuenta corriente dentro de ese banco. CC son dos dígitos de control el primero es el dígito de control de correspondiente al número de diez dígitos 00BBBBSSSS aplicando el cálculo del ejercicio 7. El segundo es el dígito de control correspondiente al número XXXXXXXXXX. Los dígitos de control KK son específicos del IBAN y se calcula añadiendo al final del número de cuenta (los 20 dígitos BBBBSSSSCCXXXXXXXXXX) seis dígitos más, dos por cada letra que representa al país y dos ceros. A cada letra le corresponde un par de dígitos: a la A le corresponde el 10, a la B el 11, y así sucesivamente hasta la Z, que tiene el 35. Luego el número resultante es BBBBSSSSCCXXXXXXXXXX142800 si fuera el caso de ES (E=14, S=28). Se calcula el resto de dividir el número resultante entre 97. KK = 98 – el resto calculado. Los dígitos de control son dos, por lo que, si el resultado del cálculo anterior tiene un dígito, hay que añadir un dígito por delante. Por ejemplo, si se introduce Banco 19 (con cuatro cifras, el 0019 – Deutsche Bank) sucursal 20 (0020) u cuenta 1234567890, el código de control del banco y sucursal (0000190020) sería 9 y el de la cuenta (12345678920) sería 6 resultado en el código completo (con blancos para aclarar) 0019 0020 **96** 1234567890. Los códigos de control se calcularían añadiendo al número 142800 al final (obteniendo 001900209612345678901428 00), obteniendo el resto de dividirlo entre 97 (32) y restándolo de 98 (98-32=66), siendo el IBAN resultante “**ES66** 0019 0020 9612 3456 7890”.